

### **Remarks**

Applicants respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. The specification has been amended to address a minor typographical mistake. Claims 1 and 3-20 are pending. Claims 1, 11, 15, and 19 are independent. No claims have been allowed. Claims 1-20 are rejected. These rejections are respectfully traversed. Claim 2 has been cancelled. Claims 1 and 3-18 have been amended, all for reasons not necessarily related to patentability (e.g., clarifying that the claims recite computer program products). The amendments herein do not necessarily narrow the claims.

### ***Information Disclosure Statement***

The Action states that NPL documents for the Hoare, Milner, and Roscoe references cited in the Information Disclosure Statement filed on June 7, 2004, have not been provided. Applicants respectfully note that the four references are textbooks that were mailed with the IDS. The books are indicated as received in an artifact sheet in PAIR. In the interest of expediting prosecution, however, soft copies of the four references are filed herewith. Applicants request that these four references be considered by the Examiner.

### ***Claim Objections***

The Action objects to claim 2 as being of improper dependent form. Applicants respectfully submit that claim 2 has been cancelled, rendering the objection moot. Therefore, Applicants respectfully request that the objection be removed.

### ***Claim Rejections under § 101***

The Action rejects claims 1-20 under 35 U.S.C. § 101 as being directed to non-statutory subject matter. These rejections are respectfully traversed. Claims 1 and 3-18 have been amended to recite “A computer program product” and claim 2 has been cancelled. Accordingly, Applicants respectfully submit that the rejections of claims 1 and 3-18 should be withdrawn. Claims 19 and 20 are directed to one or more computer-readable media having encoded thereon a data structure. A memory storing data in a particular data structure is patentable. *See, e.g., In re Lowry*, 32 F.3d 1579 (Fed. Cir. 1994) . Accordingly, Applicants respectfully submit that the rejections of claims 19 and 20 should be withdrawn.

### ***Claim Rejections under § 112***

The Action rejects claims 1-20 under 35 U.S.C. § 112, second paragraph, as being indefinite. These rejections are respectfully traversed. Claim 11 has been amended for reasons of clarity (*e.g.*, claim 11 now recites “analyzing” instead of “evaluating”).

Regarding claim 1, which recites in part “generating the partial procedure summary of the procedure,” Applicants respectfully direct the Examiner’s attention to the specification at, for example, page 9, lines 7-20:

#### ***Example 9 – Exemplary Partial Procedure Summaries***

In any of the examples, a procedure can be summarized by a plurality of partial procedure summaries. Partial procedure summaries can resemble procedure summaries and can have the same properties. However, partial procedure summaries can be used to accurately model multithreaded software.

FIG. 5 shows an exemplary arrangement 500 in which a plurality of partial summaries 530A-530N have been generated for a procedure 512. The partial procedure summaries 530A-530N can summarize less than all of the procedure 512 (*e.g.*, summarize up to a location in the middle of the procedure 512, rather than summarizing from beginning to end).

In combination, the partial procedure summaries 530A-530N can model execution of the entire procedure and accurately model state of the software 512 during

multithreaded execution, even if the procedure 512 is subject to interruption by other threads.

Regarding claim 4, which recites in part “responsive to determining the modeled state fails the indicated state invariant, indicating that a programming flaw is present in the software,” Applicants respectfully direct the Examiner’s attention to the specification at, for example, page 7, lines 13-17:

For example, one such mechanism is a specified invariant. An example implementation of a specified invariant is called an “assert.” The assert indicates a condition (i.e., and assertion) that is believed to be invariant at a particular location within the software. If the condition is false for any possible set of execution paths, a programming flaw is indicated. In such a case, the assert is said to have failed.

Regarding claim 5, which recites in part “associating an initial location and a resulting location within the procedure with the partial procedure summary,” Applicants respectfully direct the Examiner’s attention to the specification at, for example, page 12, lines 16-27:

***Example 17 – Exemplary Inclusion of Procedure Location in Summary***

In any of the examples described herein, a procedure summary can include one or more indications of a location within the procedure. For example, the procedure summary can indicate an initial location and a resulting location in the procedure. In such a case, the summary summarizes actions for the procedure starting at the initial location and ending at the resulting location. In some cases, execution may loop, so it is possible for a resulting location to be before the initial location in the code. Such locations are sometimes called the “program counter” because they reflect a modeled value of a program counter executing the procedure.

Inclusion of the initial and resulting locations can help in piecing together partial summaries. In such a case, the initial and resulting locations may not correspond to the beginning or end of the procedure being summarized.

Regarding claim 6, which recites in part “consulting a procedure summary comprising the partial procedure summary when the procedure is encountered during the reachability analysis,” Applicants respectfully direct the Examiner’s attention to the specification at, for example, page 8, line 4, to page 9, line 5:

***Example 8 – Exemplary Method for Analyzing Multithreaded Software via Reachability Analysis and Procedure Summaries***

FIG. 3 shows an exemplary method 300 using reachability and procedure summaries to analyze multithreaded software.

At 312, reachability analysis of the multithreaded software begins. For example, reachability analysis can include modeling execution of the software by determining the possible execution paths of software and systematically exploring the state of the software during such execution paths.

At 314, when a procedure is encountered, a summary for the procedure is generated. For example, partial procedure summaries can be calculated by software.

At 316, the summary is returned for consultation by the reachability analysis. The reachability analysis can then continue. For example, the reachability analysis can determine possible execution paths within the procedure and use the procedure summary to explore possible states. If the procedure is encountered again during reachability analysis, the procedure summary can be reused rather than re-generated.

Regarding claim 10, which recites in part “the plurality of actions atomically modelable with respect to multithreaded execution of the software is a proper subset of the plurality of actions of the procedure,” Applicants respectfully submit that the term “proper subset” is very clear, and also respectfully direct the Examiner’s attention to the specification at, for example, page 10, line 18, to page 11, line 6:

***Example 12 – Exemplary Partial Procedure Summaries***

FIG. 8 shows exemplary partial procedure summaries based on a series of actions of a procedure. In the example, a series of actions 812 of a procedure has been divided into two or more subsets of actions 820A-820N. In the example, the subsets of actions are atomically modelable with respect to multithreaded execution of the procedure.

Atomic modelability can include the property that the actions can be modeled together without regard to possible interleaved actions from other threads. The actions can be deemed to have occurred one after the other without interruption by other threads. For example, a set of actions that are atomically modelable may be executable as a set. Regardless of whether actions from other threads execute during execution of the set, the resulting state will be the same. The software state can thus be accurately modeled by a procedure summary constructed from such actions even if the procedure is subject to interruption by other threads.

Based on the respective subsets of actions 820A-820N, respective partial procedure summaries 830A-830N can be generated. For example, the partial summary 830A can model the changes in state caused by execution of the actions 820A, and so on.

Regarding claim 11, which recites in part “analyzing actions of the multithreaded software” and “the procedure summaries model states of the multithreaded software for

multithreaded execution of the multithreaded software,” Applicants respectfully direct the Examiner’s attention to the specification at, for example, page 8, line 20, to page 9, line 5 (copied above) and at page 9, lines 7-20 (copied above).

Regarding claim 15, which recites in part “a model checker operable to analyze a model of the multithreaded software,” Applicants respectfully direct the Examiner’s attention to the specification at, for example, page 5, lines 9-22:

***Example 1 – Exemplary System for Analyzing Multithreaded Software via Procedure Summaries***

FIG. 1 is a block diagram representing an exemplary system 100 for analyzing multithreaded software via procedure summaries. In the example, the system takes multithreaded software 112 as input. A model checker 120 analyzes the multithreaded software 112. For example, the model checker 120 can model execution of multithreaded software 112 to detect programming flaws.

In practice, the multithreaded software 112 can be first converted into a model 122, which can represent the multithreaded software 112 in a form appropriate to facilitate modeling. In the example, the model 122 can include one or more procedure summaries 124.

The model checker 120 can generate results 130. For example, the model checker can indicate whether the multithreaded software 112 contains programming flaws and information about such flaws.

Regarding claim 19, which recites in part “[o]ne or more computer-readable media,” Applicants respectfully submit that a greater-than-one phrasing is definite, and also respectfully direct the Examiner’s attention to the specification at, for example, page 13, lines 2-6:

A variety of formats can be used to represent a procedure summary (e.g., having the partial procedure summaries 530A-530N of FIG. 5) of multithreaded software. Any of the formats can be stored as a data structure on one or more computer readable media.

Accordingly, Applicants respectfully request that the rejections of claims 1-20 under 35 U.S.C. § 112 be withdrawn.

*Cited Art*

Tyrrell *et al.*, “CSP Methods for Identifying Atomic Actions in the Design of Fault Tolerant Concurrent Systems” (hereinafter Tyrrell).

*Patentability of Claims 1-20 over Tyrrell under § 102(b)*

The Action rejects claims 1-20 under 35 U.S.C. § 102(b) as being anticipated by Tyrrell. These rejections are respectfully traversed. Applicants respectfully submit that the claims in their present form are allowable over the cited art. For a 102(b) rejection to be proper, the cited art must show each and every element as set forth in a claim. (*See* MPEP § 2131.01.) However, the cited art does not so show. For example, with respect to claim 1, Tyrrell does not show generating the partial procedure summary of the procedure from the plurality of the actions atomically modelable with respect to multithreaded execution of the multithreaded software.

*Claims 1 and 3-10*

Claim 1 is directed to a computer program product embodied on a computer-readable medium and comprising code that, when executed, causes a computer to perform a method of generating a partial procedure summary of a procedure of multithreaded software, wherein the procedure performs a plurality of actions when executed, and recites:

identifying a plurality of the actions as atomically modelable with respect to multithreaded execution of the procedure; and

generating the partial procedure summary of the procedure from the plurality of the actions atomically modelable with respect to multithreaded execution of the multithreaded software (emphasis added)

For example, FIG. 6 of the present application shows an exemplary method for generating partial procedure summaries, as described at page 9, line 22, to page 10, line 2. FIG. 5 shows an exemplary arrangement in which a plurality of partial summaries have been generated for a procedure, as described at page 9, lines 7-20. FIG. 8 shows exemplary partial procedure summaries based on a series of actions of a procedure that has been divided into two or more subsets of N actions, as described at page 10, line 18, to page 11, line 6. In the described example, the subsets of actions are atomically modelable with respect to multithreaded execution of the procedure.

*Tyrrell does not show generating the partial procedure summary of the procedure from the plurality of the actions atomically modelable with respect to multithreaded execution of the multithreaded software.* In its rejection of claim 1, the Action relies on various passages in Tyrrell; however, these passages are understood to describe an expansion resulting from a trace evaluation, not “generating the partial procedure summary of the procedure from the plurality of the actions atomically modelable with respect to multithreaded execution of the multithreaded software” as recited in claim 1.

For example, the Action relies upon Tyrrell at page 633, including Figs. 3 and 4. During its discussion of a trace evaluation, Tyrrell describes a “[s]imple example with three parallel processes” (*see* Fig. 3). However, one of ordinary skill in the art could not be expected to surmise the claimed arrangement of “generating the partial procedure summary of the procedure from the plurality of the actions atomically modelable with respect to multithreaded execution of the multithreaded software” from the mere discussion of a trace evaluation or a “[s]imple example with three parallel processes.”

Applicants fail to see how a trace evaluation or a “[s]imple example with three parallel processes” would lead one of ordinary skill in the art to the claimed arrangement, which involves “generating the partial procedure summary of the procedure from the plurality of the actions atomically modelable with respect to multithreaded execution of the multithreaded software,” as recited in claim 1.

Since the cited reference does not show all of the elements recited in claim 1, Applicants believe the claim is not subject to a 102(b) rejection and request the rejection be withdrawn.

Thus, Applicants respectfully submit that claim 1 and its dependent claims, 3-10, are allowable over the cited art.

#### *Claims 11-14*

Claim 11 is directed to a computer program product embodied on a computer-readable medium and comprising code that, when executed, causes a computer to perform a method of modeling multithreaded software, and recites:

analyzing actions of the multithreaded software; and  
based on the analyzing, generating a plurality of procedure summaries for the multithreaded software;

wherein the procedure summaries model states of the multithreaded software for multithreaded execution of the multithreaded software (emphasis added).

*Tyrrell does not show generating a plurality of procedure summaries for the multithreaded software.* In its rejection of claim 11, the Action relies on various passages in Tyrrell; however, these passages are understood to describe an expansion resulting from a trace



evaluation, not “generating a plurality of procedure summaries for the multithreaded software,” as recited in claim 11.

For example, the Action relies upon Tyrrell at page 633, including Figs. 3 and 4. During its discussion of a trace evaluation, Tyrrell describes a “[s]imple example with three parallel processes” (*see* Fig. 3). However, one of ordinary skill in the art could not be expected to surmise the claimed arrangement of “generating a plurality of procedure summaries for the multithreaded software” from the mere discussion of a trace evaluation or a “[s]imple example with three parallel processes.”

Applicants fail to see how a trace evaluation or a “[s]imple example with three parallel processes” would lead one of ordinary skill in the art to the claimed arrangement, which involves “generating a plurality of procedure summaries for the multithreaded software,” as recited in claim 11.

Since the cited reference does not show all of the elements recited in claim 11, Applicants believe the claim is not subject to a 102(b) rejection and request the rejection be withdrawn.

Thus, Applicants respectfully submit that claim 11 and its dependent claims, 12-14, are allowable over the cited art.

#### *Claims 15-18*

Claim 15 is directed to a computer program product embodied on a computer-readable medium and comprising code that, when executed, causes a computer to implement a system for modeling multithreaded software, and recites:

a model checker operable to analyze a model of the multithreaded software, the model checker comprising:

a model of the software, wherein the model comprises a plurality of procedure summaries modeling states of the software during multithreaded execution of the multithreaded software (emphasis added).

For example, FIG. 1 of the present application is a block diagram representing an exemplary system for analyzing multithreaded software via procedure summaries, as described at page 5, lines 11-19. In the example, a model checker analyzes the multithreaded software. For example, the model checker can model execution of multithreaded software to detect programming flaws. In practice, the multithreaded software can be first converted into a model, which can represent the multithreaded software in a form appropriate to facilitate modeling.

*Tyrrell does not show a model checker operable to analyze a model of the multithreaded software.* In its rejection of claim 15, the Action relies on various passages in Tyrrell; however, these passages are understood to describe an expansion resulting from a trace evaluation, not “a model checker operable to analyze a model of the multithreaded software,” as recited in claim 15.

For example, the Action relies upon Tyrrell at page 633, including Figs. 3 and 4. During its discussion of a trace evaluation, Tyrrell describes a “[s]imple example with three parallel processes” (*see* Fig. 3). However, one of ordinary skill in the art could not be expected to surmise the claimed arrangement of “a model checker operable to analyze a model of the multithreaded software” from the mere discussion of a trace evaluation or a “[s]imple example with three parallel processes.”

Applicants fail to see how a trace evaluation or a “[s]imple example with three parallel processes” would lead one of ordinary skill in the art to the claimed arrangement, which involves “a model checker operable to analyze a model of the multithreaded software,” as recited in claim 15.

Since the cited reference does not show all of the elements recited in claim 15, Applicants believe the claim is not subject to a 102(b) rejection and request the rejection be withdrawn.

Thus, Applicants respectfully submit that claim 15 and its dependent claims, 16-18, are allowable over the cited art.

#### *Claims 19-20*

Claim 19 is directed to one or more computer-readable media having encoded thereon a data structure, and recites:

a plurality of state pairs representing a procedure summary for multithreaded software, wherein at least one of the state pairs comprises an initial state and a resulting state indicating a state after execution of actions modeled by the procedure summary, wherein the procedure summary models multithreaded execution of the multithreaded software (emphasis added).

*Tyrrell does not show a plurality of state pairs representing a procedure summary for multithreaded software.* In its rejection of claim 19, the Action relies on various passages in Tyrrell; however, these passages are understood to describe an action diagram, not “a plurality of state pairs representing a procedure summary for multithreaded software,” as recited in claim 19.

For example, the Action relies upon Tyrrell at page 630, including Fig. 2. With respect to Fig. 2, Tyrrell describes “(a) an action diagram, (b) a process-recovery diagram, and (c) the action diagram dual of [the process-recovery diagram].” However, one of ordinary skill in the

art could not be expected to surmise the claimed arrangement of “a plurality of state pairs representing a procedure summary for multithreaded software” from the mere discussion of “(a) an action diagram, (b) a process-recovery diagram, and (c) the action diagram dual of [the process-recovery diagram].”

Applicants fail to see how “(a) an action diagram, (b) a process-recovery diagram, and (c) the action diagram dual of [the process-recovery diagram]” would lead one of ordinary skill in the art to the claimed arrangement, which involves “a plurality of state pairs representing a procedure summary for multithreaded software,” as recited in claim 19.

Since the cited reference does not show all of the elements recited in claim 19, Applicants believe the claim is not subject to a 102(b) rejection and request the rejection be withdrawn.

Thus, Applicants respectfully submit that claim 19 and its dependent claim 20 are allowable over the cited art.

#### ***Request for Interview***

If any issues remain, the Examiner is formally requested to contact the undersigned attorney prior to issuance of the next Office Action in order to arrange a telephonic interview. It is believed that a brief discussion of the merits of the present application may expedite prosecution. Applicants submit the foregoing formal Amendment so that the Examiner may fully evaluate Applicants’ position, thereby enabling the interview to be more focused.

This request is being submitted under MPEP § 713.01, which indicates that an interview may be arranged in advance by a written request.

*Conclusion*

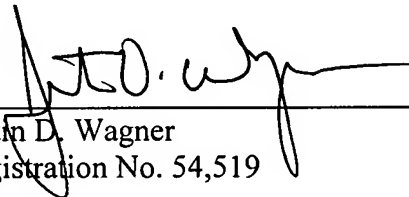
The claims in their present form should now be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600  
121 S.W. Salmon Street  
Portland, Oregon 97204  
Telephone: (503) 595-5300  
Facsimile: (503) 595-5301

By

  
Justin D. Wagner  
Registration No. 54,519